 ISBN: 978-989-95618-4-7

Ontology Driven Knowledge Extraction System with Application in e-Government

Mário Rodrigues[1], Gonçalo Paiva Dias[2], and António Teixeira[3]

[1] ESTGA/IEETA, University of Aveiro, Portugal
[2] ESTGA/GOVCOPP, University of Aveiro, Portugal
[3] DETI/IEETA, University of Aveiro, Portugal

Abstract. Important sources of information are originally created in natural language. To make that knowledge computer processable it is necessary to understand the structure of natural languages, by adding lexical and syntactic information; to have a rich representation to encode the knowledge of sentences, like ontologies; and to develop algorithms to bridge the gap between natural languages and computer processable representations. In this paper we present the architecture, modules and results of a prototype that uses an ontology to represent the world concepts and their relationships, and also to guide the process of extracting information from natural language documents.
The system was tested using minutes of Portuguese municipalities' meetings. Initial results are presented for three topics of municipalities' affairs: the subsidies granted, the building permits requested, and the existing protocols with other institutions.

Keywords: entities and relations extraction, ontology, e-government.

1 Introduction

Important sources of information are originally created in natural language (NL) documents - English, Portuguese, etc. Although stored in computers, these documents do not contain a formal indication about the data types they contain. This lack of formal indication prevents the information to be manipulated to meet user's specific needs when accessing/querying/searching it [1].

From the information representation perspective "Knowledge bases play an increasingly important role in enhancing the intelligence of Web and enterprise search, as well as in supporting information integration. Today, most knowledge bases... are very cost intensive to keep up-to-date as domains change" [2].

To bridge the gap between natural languages and computer processable representations it is necessary to understand the structure of natural languages by adding lexical and syntactic information, and to have a rich representation to encode the knowledge of those sentences. Moreover, it is necessary to develop methods that allow changing the knowledge domain without re-programming/re-engineering the system in order to keep a low operational cost.

In this paper we present a system able to create semantic information from natural language unstructured documents using natural language processing algorithms, integrating open source software, and using external sources of information as Google Maps and Geo-Net-PT01 [3]. The system can be used with any ontology. To achieve this important feature, the knowledge domain is represented by an ontology that also guides the process of extracting information from the natural language documents.

The system was tested with municipalities' that contain information that would benefit users if made available in searchable knowledge bases. E-government would benefit from systems able to integrate several sources of information and able to understand unstructured documents. These particular documents are important because municipalities are often the closest point of service for citizens and enterprises, and the minutes record the decisions of the municipalities.

The remaining of the paper starts by discussing the related work. Section 2 starts with an overview of the developed system and continues by elaborating on each of the three parts that compose it. In Section 3 the results are presented and discussed. The paper ends with the conclusions in Section 4 and the acknowledgments in Section 5.

1.1 Related Work

Several projects were dedicated to the task of scalable, domain independent information extraction. DBPedia [2] is a knowledge base created by extracting information from Wikipedia infoboxes and using its structure to infer the semantics. A similar approach was followed to create Yago knowledge base [4]. In addition, a set of axiomatic rules was defined to improve the information extraction precision, and WordNet [5] was used to disambiguate word meanings. These knowledge bases were created without any natural language processing.

Kylin system [6] uses Wikipedia infoboxes information to train statistical classifiers that later extract information from natural language texts. The features used in training are part-of-speech (POS) tags and surface features (position in the sentence, capitalization, presence of digits or special characters, etc.). It does not use syntactic information.

Other state of the art systems did not used Wikipedia as a knowledge resource. TextRunner [7] aims to extract all instances of all meaningful relations from web pages. It constructs the ontology from the corpus. It does not control if the ontological relations are well defined, and does not disambiguate words to entities. KnowItAll [8] uses manually specified examples that express a set of relationships (e.g. friend(John,Peter)). It uses those facts to find textual patterns that could possibly express the relations (e.g. John is a friend of Peter). The textual patterns are used to train a set of predefined information extraction templates. Leila [9] has further improved KnowItAll method by using both examples and counterexamples as seeds, in order to generate more robust patterns, and using deep syntactic analysis to generate the information extraction patterns. Our approach is based on Leila because the deep syntactic analysis helps capturing information in long (and complex) sentences.

The research activity done so far in e-government is usually centered in solving problems as interoperability and service integration. In such projects it is usually considered that the information is already in the system, whether placed by human operators or using existing databases (e.g. OneStopGov and Access-eGov). The problem of automatic acquisition of information from natural language government documents still needs more attention [1].

2 Developed System

The developed system is organized in 3 main modules:

Natural Language Processing - includes document content processing technologies module to retrieve structured information from NL texts. Its named natural language processing (NLP) because it is based on technologies from the NLP area;

Knowledge Domain - a module that haves tools to define the data semantics and associate it with samples of the NLP module output;

Semantic Extraction and Integration - it contains integration tools that learn the syntactic/semantic associations and applies them to transform the information on meaningful semantic knowledge. It also complements that knowledge with external structured sources.

The result is the semantic information can be queried and accessed instead (or in complement) of the original documents (see Fig. 1). The system was built reusing open source software - some of it adapted to work with Portuguese - to take advantage of state of the art approaches and software. Specific software was developed to integrate the reused software in a coherent system.

The system operates in two modes: training mode, and runtime mode. In training mode, the NLP module extracts and processes a sample of NL text. It adds information (tags) producing an enriched version containing named entities, POS and syntactic information. Then the system manager (a human) defines the semantics of the system by creating/adapting/importing an ontology using an ontology editor that features a graphical user interface (GUI). Using another knowledge domain (KD) tool, the system manager associates ontology classes and relations to the enriched text sample, creating examples of correspondences between syntactic structures and semantic concepts. The training ends with the semantic relation extraction part of the semantic extraction and integration (SEI) module training semantic models that associate syntactic structures to semantic concepts based on the examples given by the system manager.

In runtime, NL text is enriched by the NLP module just like in the training stage. The output of this module is passed to the SEI module that uses the trained semantic models to identify semantic concepts in the enriched text. If information is missing (according to the ontology) the system can look up external structured sources in order to complete the information. The semantic information extracted from the text and from external sources is stored in a knowledge base.

 ISBN: 978-989-95618-4-7

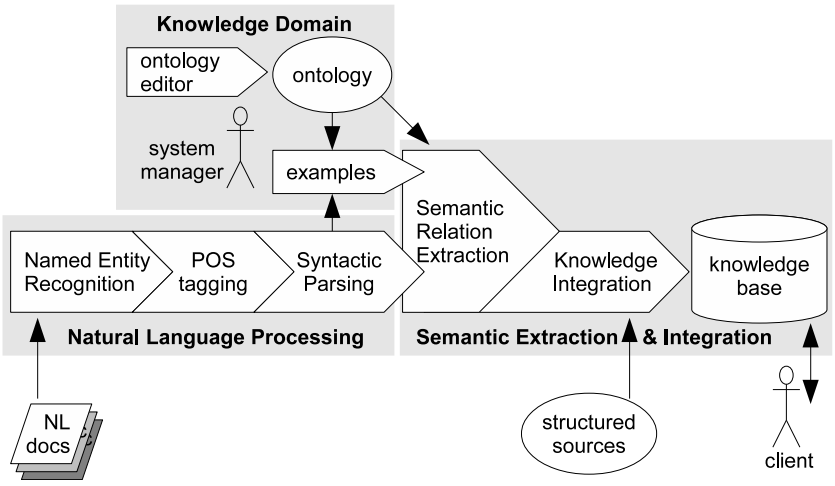


Fig. 1. The system is organized in three modules (*Natural Language Processing (NLP), Knowledge Domain (KD) and Semantic Extraction and Integration (SEI)*). The NLP module enriches NL text with POS tags and syntactic structures. The KD defines the system semantics and provides examples of syntactic/semantic correspondences. The SEI module learns to extract semantic information based on the KD examples, applies the learned models to all texts, integrates information from external structured sources, and stores the data according to the defined semantics.

The following sections will elaborate on each module. The explanations are illustrated using the fragment of a minute presented in the first row of Table 1.

2.1 Natural Language Processing Module

This section describes the components of the NLP module of the system: named entity recognition (NER), POS tagging, and syntactic parsing. NER is the task of locating and classifying atomic elements in the text into predefined categories such as the names of persons, organizations, locations and so on [10]. POS tagging is the task of assigning parts of speech to each word (and other token), such as noun, verb, adjective, etc. [11]. Syntactic parsing is the task of analyzing a text made of a sequence of tokens to determine its structure with respect to a given grammar [12].

NER is performed by a system developed for Portuguese named Rembrandt. Rembrandt uses Wikipedia as a raw knowledge resource and its document structure to classify all kind of named entity (NE) in the text according to the Second HAREM directives [13, 14]. Unclassified NEs are collected to be classified using another strategy. The strategy is to query the Google Maps API about the location of the NE and, if a location is retrieved, the NE is classified as an entity with a fixed physical location. In this case, the entity is marked as having latitude and longitude which can be an organization (enterprise or institution headquarters),

 ISBN: 978-989-95618-4-7

Table 1. Example fragment and output of the system modules.

Fragment:	"... atribuir os seguintes apoios financeiros: ... À ARCEL - Associação Recreativa e Cultural de Espinhel, um subsídio no valor de 8.640,00€, destinado a apoiar a execução do Plano Anual e a Escola Artística"			
Translat.:	"... to award the following financial aid: ... To ARCEL - Associação Recreativa e Cultural de Espinhel, a subsidy amounting to €8,640.00, to support the implementation of the Annual Plan and the Art School".			
NLP out:	1	À_Arcel	prop	0 UTT
	2	–	punc	1 PUNC
	3	Associação_Recreativa...	prop	1 N<PRED
	4	,	punc	1 PUNC
	5	um	art	6 >N
	6	subsídio	n	1 N<PRED
	7	em	prep	6 N<
	8	o	art	9 >N
	9	valor	n	1 N<PRED
	10	de	prep	9 N<
	11	8.640,00€	num	9 N<
AKTive M.:	example Subsidy:subsídio requester Organization:À_Arcel example Subsidy:subsídio moneyAmount moneyAmount:8.640,00			
KB entry:	<owl:NamedIndividual rdf:about="<URI>#s_8.64000eur"> <rdf:type rdf:resource="<URI>#Subsidy"/> <moneyAmount>8.64000eur</moneyAmount> <requester rdf:resource="<URI>#a_arcel"/> <terms:isReferencedBy rdf:resource="<URI>#acta_6902"/> </owl:NamedIndividual>			

a place (physical or human), or an event that happens always in the same place. Words belonging to a NE are grouped together to be treated as single tokens in the rest of the processing pipeline.

The POS tagging is performed by TreeTagger. It annotates text with POS and lemma information and has been successfully used to tag several natural languages including Brazilian Portuguese. The tagger was trained for European Portuguese using the corpus Bosque v7.3. Bosque is a subset of Floresta (a publicly available Treebank for Portuguese), fully revised by a linguistic team, that contains about 185,000 words [15]. This particular version of Bosque was selected because it was also necessary to train the syntactic parser, and Bosque v7.3 is the only Portuguese corpus usable to train the chosen syntactic parser. Since TreeTagger training process uses separate files for corpus and lexicon, it was possible to extend the lexicon using the computational lexicon LABEL-LEX-sw that comprises more than 1,500,000 inflected word forms, automatically generated from a lexicon of about 120,000 lemmas [16].

The syntactic parsing is done with a data-driven dependency parser named MaltParser [17]. MaltParser was already successfully used to parse several natural languages as English, French, Greek, Swedish, and Turkish. The parsing algorithm chosen was the same of the Single Malt system [17] and the parsing model was induced with the Bosque v7.3 used in the Tenth Conference on Computational Natural Language Learning (CoNLL-X). This particular version was selected because it's available for Portuguese in the CoNLL-X format, the format accepted by MaltParser.

After the parsing step, the POS tag assigned to each NE is replaced by the class assigned by Rembrandt to that NE. This allows taking advantage of the information given by Rembrandt about the class of the NE. The output of this module is presented in the third row of the Table 1. Tags were assigned to all words that are not NE. Word *um* was tagged as an article (*art*); word *subsídio* was tagged as a noun (*n*) and so on and so forth. The result of the syntactic parser can be seen in the fourth and fifth columns. The fourth column denotes the dependency of the word (0 has no dependency; other number indicates the dependency of that word number) and the fifth columns denotes the type of dependency. For instance, the word *um* which is the word number 5 (see first column) depends of the word number 6 (see fourth column of word *um*) which is the word *subsídio*. Moreover it is a dependency to a noun (*>N* in the fifth column of word *um*).

2.2 Knowledge Domain Module

The knowledge domain representation is done with an ontology. An ontology is formally defined as an explicit specification of a shared conceptualization [18]. The ontology is defined in web ontology language (OWL) and was created using Protégé-OWL v4.1. It combines the ontologies Friend of a Friend (FOAF), Dublin Core, World Geodetic System (1984 revision), and GeoNames (full version). A new class named *ExecutiveSubject* was added to handle subjects relative to municipalities. *ExecutiveSubject* is a subclass of the top level class *Thing* and

has seven subclasses or specializations (see Tables 2). Six object relations define the type of relations that the instances of class *ExecutiveSubject* can have (Table 3). Disjunctions between classes and/or relations were also defined in the ontology. For instance, *Allotment* is disjoint with *Protocol* because protocols are not allotments and vice-versa, but *ConstrProcess* is not disjoint with *BuildPermit* because they both refer to construction and some overlap is possible.

Table 2. Subclasses of the class *ExecutiveSubject*.

Class name	Class description
Allotment	Permissions to perform land allotment or change previous allotments.
BuildPermit	Relative to construction contracts, public or private, already in execution.
ConstrProcess	Announces relative to generic public construction processes: beginning of works, changes in budgets, expropriations, etc.
Exemption	Requests for municipal fees exemptions.
Protocol	Protocols signed with institutions like schools or local clubs.
PublicCalls	Announcements of public contracts to acquire equipment or build some facility.
Subsidy	Granted/requested subsidies or allowances.

Table 3. Types of relations for *ExecutiveSubject*.

Relation name	Relation description
deliberation	The outcome.
identifier	The unique identifier given by services.
moneyAmount	Any money amount involved in the process.
motivation	The motive.
place	The address of the: construction/allotment, institution that signed the protocol, or entity that requests an exemption or subsidy.
requester	The entity or entities, excluding the municipality, that is/are involved in the process.

The association between text samples and ontology classes and relations is made using AKTive Media ontology based annotator [19]. The human annotator starts by highlighting parts of text and assigns them an ontology class. Then it is possible to select an ontology relationship with another part of the text (see Fig. 2). Individuals without ontological relationships are ignored to prevent annotation mistakes.

The result of this step is a text file containing a binary relation per line. Each line starts with the keyword *example* followed by the subject of the relation coded as *subject_ class:subject_ text*. Then follows the relation type *relation_ name* and the object of the relation coded the same way the subject of the relation (fourth row of Table 1).

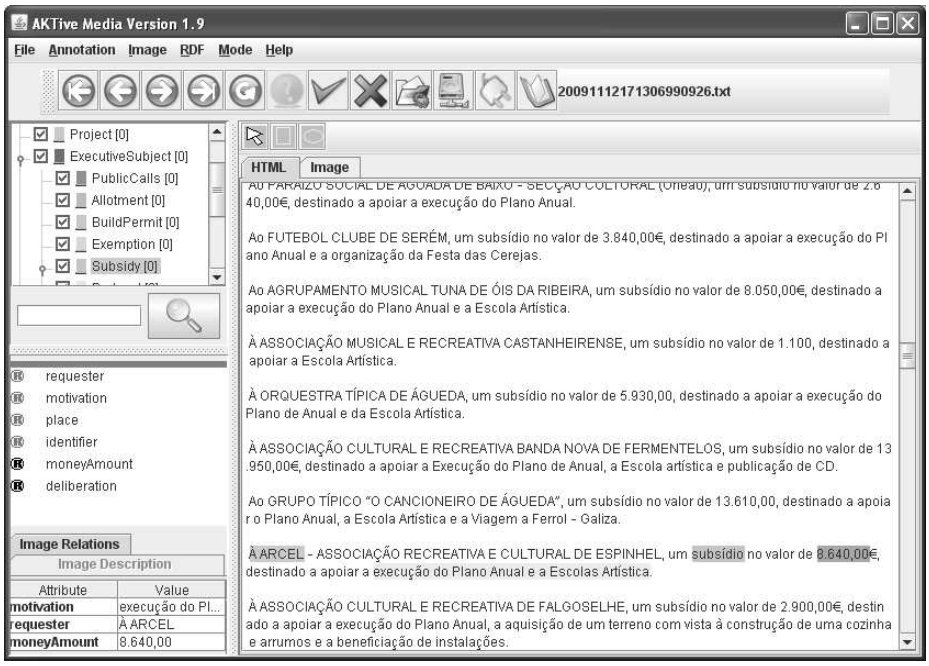


Fig. 2. Screen shot of AKTive Media interface. The top left pane shows the ontology classes. Below, after the search widget, a pane shows all relations of the selected class. The large pane at the right shows two words highlighted. Clicking on top of a highlighted word appears all its attribute/value pairs in bottom left pane.

2.3 Semantic Extraction and Integration Module

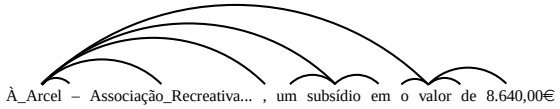
This module was inspired in LEILA [9]. LEILA requires a set of functions to decide if a pair of words is: an example - if it belongs to a list of examples; a counterexample - if it is incompatible with the examples; a candidate - if it obeys to some criteria and is neither an example nor a counterexample; or nothing - if is none of the previous and should be ignored. This classification scheme was kept from the original implementation and a different algorithm was used for extracting semantic relations. The reason for the change is that LEILA needs a function for each relation and we want a single procedure to handle all and any relations created in the ontology. This eliminates the need to reprogram the function(s) if the ontology is changed.

The training procedure of LEILA generates one k-nearest neighbor (K-NN) classifier for each decision function. Our training procedure generates a set of K-NN classifiers - one for each ontology class and relation - that are used in runtime to detect semantic information in syntactic structures. There are two types of classifiers: relation classifiers and entity-of classifiers. Both use planar graphs that code the enriched text sentences according to the dependencies given

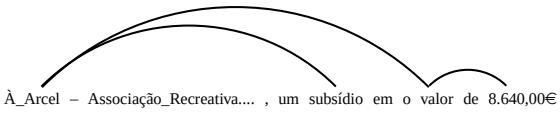
by the syntactic parser. The nodes of the graphs are the words of the sentence and the edges are the (labeled) dependencies (see Fig. 3(a)).

Relation classifiers assess subject/object pairs based on the shortest graph path (the bridge) between the elements of the pair. These bridges are produced and stored the same way as in LEILA [9]. Two bridges are regarded as equivalent if they have the same sequence of nodes and edges, although nouns and adjectives are allowed to differ. Fig. 3(b) depicts the bridge used by the relation classifier to associate *subsídio* and *8.640,00€*.

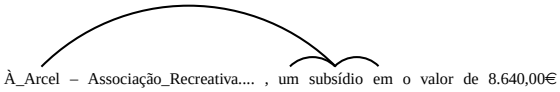
Entity-of classifiers associate subjects and/or objects to their ontological classes based on the connections between the subject/object and the other tokens of the sentence. The entity-of classifiers were specifically developed and store a collection of pairs for each subject/object. Two tokens are regarded as equivalent if they have, at least, one connection to the same lemma using the same edge, although nouns and adjectives are allowed to differ. Fig. 3(c) depicts the three pairs stored by the entity-of classifier to characterize the token *subsídio*.



(a) The full sentence graph.



(b) A bridge used by relation classifiers to associate *subsídio* and *8.640,00€*.



(c) The set of connections used by the entity-of classifier to classify *subsídio* as an element of the ontology class *Subsidy*.

Fig. 3. Syntactic structure graph associated to the fragment presented in Table 1.

Training Mode This procedure starts by reading the output file of AKTive Media annotator. Each line is broken in three facts stored in memory: *subject_ text* is individual of class *subject_ class* (for entity-of classifier); *object_ text* is individual of class *object_ class* (for entity-of classifier); and *subject_ text* has relation *relation_ name* with *object_ text* (for relation classifier). Then the set of sentence

graphs is iterated to find pairs of *subject_ text object_ text* as described in the *find examples* part of the Algorithm 1.

Because it is assumed that all relations were marked in the sample texts, the algorithm continues with the search of relation counterexamples. Relation counterexamples are searched by having relation classifiers evaluating all word pairs. All positive evaluated pairs that are not examples are added as counterexamples.

For entity-of classifiers a different strategy is followed because an entity (like a person name) can be element of a class (a person) but is not involved in any relevant relation and thus is not an example or counterexample. The ontology is loaded and the examples associated to an ontological class (given by the respective entity-of classifier) will be counterexamples of all entity-of classifiers associated with its disjoint ontological classes. This step is called entity-of classifiers information share.

Algorithm 1 Training procedure

```
{find examples...}
for all s : sentenceGraphs do
  for all p : example_pairs do
    if p ∈ s then
      relClassifier ← getRelationClassifier(p.relation)
      relClassifier.addBridgeBetween(p.subject.text, p.object.text)
      subClassifier ← getEntityOfClassifier(p.subject.class)
      subClassifier.addConnectionSetOf(p.subject.text)
      objClassifier ← getEntityOfClassifier(p.object.class)
      objClassifier.addConnectionSetOf(p.object.text)
    end if
  end for
end for
{find relation counterexamples}
{entity-of classifiers information share}
```

Runtime Mode This procedure starts with all classifiers evaluating all words pairs of all sentences' graphs. Positive evaluated pairs are collected in temporary buffers, one buffer for each classifier (see Algorithm 2).

Then all relations are loaded from the ontology (ontology relations are triples like: *subject class* has *relation* with *object class*). For each ontology relation, the buffers of the classifiers relative to that relation are loaded. Every pair that exists in the three buffers is stored in the knowledge base. Each saved pair means that the *pair subject* has *relation* with the *pair object*.

In the end, a semantic reasoner acts upon the knowledge base to verify if all information is coherent. The reasoning is performed by an open source reasoner for OWL-DL named Pellet [20]. Then the knowledge base is stored and managed by Virtuoso Universal Server. If the knowledge base in not coherent, a warning is issued and no further processing is done.

The missing information - according to the ontology - is searched in external structured sources. For instance, unknown locations of entities with a fixed place (as streets, organizations headquarters, and some events) are queried using Google Maps API. Also, the political organization of the spaces - street $\subset$ neighborhood $\subset$ city $\subset$ municipality ... - is obtained using a geographic ontology of Portugal with about 418,000 features named Geo-Net-PT01 [3]. This allows the system to display the information spatially on a map and to search and relate information by its location [21].

The last row for Table 1 shows an entry of the knowledge base relative to the example. It is visible an *owl:NamedIndividual* of type *Subsidy* with some *moneyAmout*, with property *assignedTo a_ arcel* and *isReferencedBy acta_ 2009...* (in Portuguese "acta" means minute). This entry defines a subsidy. Other entries (not showed) define the minute *acta_ 2009...* and the NE *a_ arcel*.

Algorithm 2 Runtime procedure

```
{find candidates...}
for all c : classifiers do
  b ← getResultBuffer(c.classifierType)
  for all s : sentenceGraphs do
    for all p : s.allPairs do
      if c.evaluate(p) > 0 then
        b.add(p)
      end if
    end for
  end for
end for
{filter candidates}
{reasoning and integration}
```

3 Results

Experiments were conducted to extract information about three topics of municipalities' public documents: subsidies granted, building permits requested, and protocols with other institutions.

A web crawler obtained all documents available in seven Portuguese municipalities' websites. Two random sets of 50 documents were selected. The documents were in pdf file format and no document was is both samples. One set was manually annotated by a human and the annotations were used to train the system classifiers. The other set was used in runtime to have knowledge extracted by the system.

3.1 Use Case Evaluation

To illustrate the usefulness of this type of systems when searching information, let's consider a citizen called *Maria*. *Maria* is a frequent name in Portugal. A keyword based query to the test set returned 165 results. A semantic query about the building permits (ProcessoDeObra) applied by persons called *Maria* returned 2 results (see Table 4). Moreover, besides restricting the results by specifying the type of the searched information, it is possible to have a resume with the document where the information was found, the identifier that the municipality assigned to the process and the outcome so far. The SPARQL query made to Virtuoso is in Algorithm 3.

Algorithm 3 SPAQRL query about which persons applied a building permit.

```
PREFIX municip: <http://.../municipality.owl#>
SELECT ?person ?document ?id ?outcome
WHERE {
 ?proc rdf:type municip:BuildPermit; municip:requester ?pret.
 ?page foaf:topic ?proc; terms:title ?document.
 ?pret foaf:name ?person.
 OPTIONAL {?proc terms:identifier ?id}.
 OPTIONAL {?proc municip:deliberation ?outcome}.
 FILTER(REGEX(?person,"maria")).
}
```

Results are presented in Table 4. The first column shows (part of) the full name of the person, and the document where the information was found if in the second column. In the third and fourth columns are the municipality identifier of the building permit process and the outcome so far. The outcome *solicitar* (request) means that the municipality is requesting for more documentation.

Table 4. Status of the building permits requested by citizens who name includes *Maria*. The outcome *solicitar* (request) means the citizen needs to present missing documents.

person	document	id	outcome
maria_adel...	cm-arouca.pt_ACTA_12_2009	12/09	solicitar
maria_hele...	cm-arouca.pt_ACTA_22_2008	153/2008	solicitar

3.2 Performance Evaluation

A person evaluated the knowledge extracted by the system by reading the documents and registering which information was found or not, and which information was incorrectly extracted. Information was considered found if the system

detected one of subsidy; build permit; protocol, even if it missed some facts like the requester or the money amount involved in the transaction. Results are summarized in Table 5.

A total of 32 subsidies were found in the test set. Of those the system detected 14 and there was any false detection. Regarding building permits, from a total of 68 the system detected 67. The system wrongly marked 4 building permits that did not exist. The system also detected 8 of the 41 existing protocols and also marked one protocol that did not exist.

The global performance (precision 0.95; recall 0.63) is comparable with the performance of state-of-the-art systems: DBpedia (precision 0.86 to 0.99; recall 0.41 to 0.77), Kylin (precision 0.74 to 0.97; recall 0.61 to 0.96), and YAGO/NAGA (precision 0.91 to 0.99; recall not reported).

Table 5. Knowledge detected by the system. For each municipality's documents set are presented: the total number of correctly detected information and, between brackets, the total information in those documents. Adding to these information are ([1]) 4 building permits incorrectly extracted and ([2]) 1 protocol incorrectly extracted.

	municipality									
	a	b	c	d	e	f	g	precision	recall	F_1
subsidy	0(2)	3(3)	4(11)	1(1)	1(1)	3(14)	0(0)	1.00	0.44	0.61
build permit	3(4)[1]	13(13)	47(47)	0(0)	0(0)	4(4)	0(0)	0.94	0.99	0.97
protocol	3(4)	3(3)	0(3)	0(0)	7(24)	2(7)[2]	0(0)	0.89	0.20	0.32
total								0.95	0.63	0.76

4 Conclusions and Future Work

This article presented a conceptual model to detect and organize relevant information from unstructured NL documents. Key features of this model are the ability to accept - without any software reconfiguration - a knowledge domain defined by an ontology, and to be able to process natural language texts. When changing domain, the only necessary task is to give some examples how to detect and integrate information according to the ontology. It is also possible to extend the system for other natural languages by changing the NLP module, and without changing the other system modules. The system also features a structured information entry point to allow the inclusion of information sources that complement the information included in the NL documents.

The prototype was developed for Portuguese language. The software tools and respective setup were described and it was explained how the tools work together in order to have a complete and coherent system. The system was tested with fifty publicly available documents from seven municipalities. Results show that the system is able to acquire useful, meaningful information from those documents. The performance tests will be scaled up but for now they show that this design has a performance comparable with state-of-the-art systems.

Future work will entail improving the system efficiency, maturing the technology, and testing the system with other knowledge domains. Another interesting improvement can be the inclusion of a more intuitive, user friendly interface.

5 Acknowledgements

The authors would like to thank Ciro Martins for the careful annotation of the training document set.

References

1. M. Rodrigues, G. P. Dias, and A. Teixeira, "Human language technologies for e-gov," in *Proceeding of the WEBIST 2010, 6th International Conference on Web Information Systems and Technologies*, J. Filipe and J. Cordeiro, Eds., vol. 2. Valencia, Spain: INSTICC, April 2010.
2. C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "Dbpedia - a crystallization point for the web of data," *Web Semantics: Science, Services and Agents on the WWW*, vol. 7, no. 3, pp. 154 – 165, 2009.
3. M. Chaves, M. Silva, and B. Martins, "A geographic knowledge base for semantic web applications," in *Proc. of SBBD*, 2005.
4. F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," in *WWW '07*, 2007, pp. 697–706.
5. G. A. Miller, "Wordnet: a lexical database for english," *Commun. ACM*, vol. 38, pp. 39–41, November 1995.
6. F. Wu, R. Hoffmann, and D. S. Weld, "Information extraction from wikipedia: moving down the long tail," in *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '08. New York, NY, USA: ACM, 2008, pp. 731–739.
7. M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, January 2007.
8. O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates, "Web-scale information extraction in knowitall:(preliminary results)," in *Proceedings of the 13th international conference on World Wide Web*. ACM, 2004, pp. 100–110.
9. F. Suchanek, G. Ifrim, and G. Weikum, "Leila: Learning to extract information by linguistic analysis," in *Proc. of the ACL Workshop OLP*, 2006.
10. K. Bontcheva, B. Davis, A. Funk, Y. Li, and T. Wang, "Human language technologies," *Semantic Knowledge Management*, 2008.
11. R. Mihalcea, "Performance analysis of a part of speech tagging task," *Computational Linguistics and Intelligent Text Processing*, pp. 299–321, 2010.
12. D. Jurafsky and J. H. Martin, *Speech and Language Processing*. Prentice Hall New York, 2000.
13. N. Cardoso, "Rembrandt - reconhecimento de entidades mencionadas baseado em relações e análise detalhada do texto," in *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM*. Linguateca, 2008.
14. C. Mota and D. Santos, Eds., *Desafios na avaliação conjunta do reconhecimento de entidades mencionadas: O Segundo HAREM*. Linguateca, 2008.

15. C. Freitas, P. Rocha, and E. Bick, "Floresta sintá (c) tica: Bigger, thicker and easier," *Computational Processing of the Portuguese Language*, 2008.
16. E. Ranchhod, C. Mota, and J. Baptista, "A computational lexicon of portuguese for automatic text parsing," in *Proc. of SIGLEX99: Standardizing Lexical Resources - ACL*, 1999.
17. J. Hall, J. Nilsson, J. Nivre, G. Eryiğit, B. Megyesi, M. Nilsson, and M. Saers, "Single malt or blended? a study in multilingual parser optimization," in *Proc. of the EMNLP-CoNLL*, 2007.
18. T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, pp. 199–220, 1993.
19. A. Chakravarthy, F. Ciravegna, and V. Lanfranchi, "Cross-media document annotation and enrichment," in *Proc. 1st Semantic Web Authoring and Annotation Workshop (SAAW2006)*, 2006.
20. E. Sirin, B. Parsia, B. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Web Semantics: science, services and agents on the World Wide Web*, vol. 5, no. 2, pp. 51–53, 2007.
21. M. Rodrigues, G. P. Dias, and A. Teixeira, "Knowledge extraction from minutes of portuguese municipalities meetings," in *Proceeding of the FALA 2010 - VI Jornadas en Tecnología del Habla and II Iberian SLTech Workshop*, 2010.